(54) **Turbo code interleaver with near optimal performance**

Verschachteler für Turbo-kode mit annähernder optimaler Leistung

Dispositif d'entrelacement de code turbo à performances quasi optimales

(72) Inventors:
• **Eroz, Mustafa**
Germantown
MD 20876 (US)
• **Hammons, A., Roger, Jr.**
North Potomac
MD 20878 (JP)
• **Sun, Feng-Wen**
Germantown
MD 20874 (US)

(74) Representative: **Brunner, John Michael Owen et al**
Carpmaels & Ransford LLP
One Southampton Row
London WC1B 5HA (GB)

(56) References cited:
US-A- 5 742 612

• **EROZ M ET AL: "On the design of prunable interleavers for turbo codes", VEHICULAR TECHNOLOGY CONFERENCE, 1999 IEEE 49TH HOUSTON, TX, USA 16-20 MAY 1999, PISCATAWAY, NJ, USA,IEEE, US, vol. 2, 16 May 1999 (1999-05-16), pages 1669-1673, XP010342111, DOI: 10.1109/VETEC.1999.780687 ISBN: 978-0-7803-5565-1**
• **MARIC S V: "CLASS OF ALGEBRAICALLY CONSTRUCTED PERMUTATIONS FOR USE IN PSEUDORANDOM INTERLEAVERS", ELECTRONICS LETTERS,GB,IEE STEVENAGE, vol. 30, no. 17, 18 August 1994 (1994-08-18), XP000472069, ISSN: 0013-5194**
• **BERROU C ET AL: "NEAR SHANNON LIMIT ERROR - CORRECTING CODING AND DECODING: TURBO-CODES (1)", PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON COMMUNICATIONS (ICC),US, NEW YORK, IEEE, vol. -, 23 May 1993 (1993-05-23), XP000371240, ISBN: 0-7803-0950-2**
• **DIVSALAR D ET AL: "TURBO CODES FOR PCS APPLICATIONS", COMMUNICATIONS - GATEWAY TO GLOBALIZATION. PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON COMMUNICATIONS. SEATTLE, JUNE 18 - 22, 1995; [PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON COMMUNICATIONS (ICC)], NEW YORK, IEEE, US, vol. 1, 18 June 1995 (1995-06-18), pages 54-59, XP000532968, ISBN: 978-0-7803-2487-9**

EP 2 173 036 B1

Description

BACKGROUND OF THE INVENTION

[0001] The present invention relates to error correction in coding schemes for digital communication systems, and more particularly to design optimization for Interleavers of any size within a specified wide range used in such error correction. Even more particularly, the present invention relates to optimization of Turbo Interleavers such that smaller optimal Interleavers can be built from larger optimal Interleavers.

[0002] Interleaving is a process of reordering a sequence of symbols or bits in a predetermined manner. "Interleaver size" is equal to the size of the sequence. The apparatus performing the interleaving is referred to herein as an Interleaver.

[0003] Turbo Interleavers are interleavers used in the construction of turbo codes. In a turbo code built as a parallel concatenation of two constituent recursive convolutional codes, a Turbo Interleaver serves to re-order an input data sequence in a pseudo-random fashion prior to an encoding by a second of the constituent codes. As a result, separate encodings produced by the two constituent encoders are largely uncorrelated, which property allows them to be combined by a turbo encoder to produce a composite encoding with excellent error protection capability.

[0004] S-random Interleavers are one of the most widespread forms of turbo Interleavers.

[0005] The principle behind S-random Interleavers is to avoid mapping neighbor positions of an original input sequence to another neighbor position of the interleaved sequence within a window of size S. The design goal in S-random Interleavers is to maximize S while preserving the above principle. However, S-random Interleavers have to be re-designed every time the Interleaver size is changed and there is typically no requirement of any resemblance between the Interleavers with similar sizes.

[0006] Thus, it is desirable to have a general Interleaver design for Interleavers of any size within a set of sizes, wherein the design methodology is concise and efficient such that the same Interleaver design is near-optimal for all Interleavers within the set of sizes. It is also advantageous to have a design for building a near-optimal Interleaver that can easily be reduced to smaller-sized near-optimal Interleavers without performance degradation.

[0007] Therefore, the present invention advantageously addresses the above and other needs.

[0008] The publication by D.Divsalar and F. Pollara, "Turbo Codes for PCS Applications", proceedings of the international conference on communications, IEEE, Seattle, June 1995, pages 54-59, discloses a turbo interleaver constructed as a block interleaver with a number of rows and of columns, in which the rows and columns are permuted in order to break low-weight sequences.

[0009] The publication by S Marić, "Class of algebraically constructed permutations for use in pseudorandom interleavers", Electronics Letters, 18th August 1994, Vo.30, No. 17 discloses a class of pseudo-random interleavers based on a permutation rule using successive powers of a primitive root of a Galois field GF(q).

SUMMARY OF THE INVENTION

[0010] The present invention advantageously addresses the needs above as well as other needs by providing a method and apparatus for a turbo Interleaver which employs Interleavers of variable length employing one or more permutations.

[0011] In one embodiment, the invention is characterized as a method of interleaving blocks of indexed data of varying length. The method includes the steps of: providing a set of basic Interleavers comprising a family of one or more permutations of the indexed data and having a variable length; selecting one of the basic Interleavers based upon a desired Interleaver length L; and adapting the selected basic Interleaver to produce an Interleaver having the desired Interleaver length L.

[0012] In another variation, a method of interleaving blocks of indexed data of variable length includes the steps of: providing a family of basic Interleavers comprising "two-dimensional permutations" including computing the "two-dimensional permutations", further comprising: writing the indexed data into an Interleaver matrix having one or more rows in each of two dimensions; permuting the indexed data in one or more rows in at least one of the two dimensions to produce "constituent permutations", possibly being different from one row to another row, wherein the constituent permutations are pseudo-random permutations described by a limited number of parameters, wherein an amount of storage required for storing the limited number of parameters is less than that for storing a vector representation of the constituent permutations; reading out the data from the Interleaver matrix; selecting one of the basic Interleavers for use in encoding based upon a desired Interleaver length L; adapting the selected basic Interleaver to produce an Interleaver having the desired Interleaver length L; wherein the selecting includes:

identifying a group of the basic Interleavers having a length greater than or equal to the desired Interleaver length L; and selecting one of the basic Interleavers having a length smallest among the identified group of the basic Interleavers; wherein the adapting includes:

deleting indexed data having indices higher than required for a permutation of length L; providing an Interleaver device for interleaving blocks of indexed data, the Interleaver device further comprising a memory device for storing descriptions of the basic Interleavers; and
storing the descriptions in the memory device.

[0013]   In another embodiment, a system for interleaving and turbo encoding blocks of indexed data, of varying length, comprises: a parallel concatenation of two or more constituent encoders for recursive convolutional codes of recursion period p; and an Interleaver device coupled to the parallel concatenation for performing the steps of: accessing stored descriptions of basic Interleavers, the basic Interleavers comprising a family of one or more permutations of the indexed data and having a variable length; identifying a group of the basic Interleavers having a length greater than or equal to a desired Interleaver length L; selecting one of the basic Interleavers having a length which is smallest among the group of the basic interleaves; and adapting the selected one of the basic Interleavers to produce an Interleaver having the desired Interleaver length L.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014]   The above and other aspects, features and advantages of the present invention will be more apparent from the following more particular description thereof, presented in conjunction with the following drawings wherein:

FIG. 1 is a block diagram of hardware of a mobile communication system of a type that could be used to implement the teachings of the present invention;
FIG. 2 is a functional block diagram of a Turbo encoder which could be implemented in the system of FIG. 1;
FIG. 3 is a flow chart of steps traversed by the mobile communication system of FIG. 1 and encoding system of FIG. 2 in accordance with an embodiment of the present invention;
FIG. 4 is a diagram of performance curves of Turbo Interleavers such as shown in FIG. 2 of Size 1024 bits at Code Rate ½, using four (4) decoder iterations comparing Galois Field Interleavers to S-Random Interleavers and to Random Interleavers for Bit Error rate (BER) and Frame Error Rate (FER) performances;
FIG. 5 is a diagram of performance curves of a Turbo Interleaver such as shown in FIG. 2 of Size 1024 at Code Rate ½, using eight (8) decoder iterations comparing Galois Field Interleavers to S-Random Interleavers and to Random Interleavers for Bit Error Rate (BER) and for Frame Error Rate (FER); and
FIG. 6 is a diagram of performance curves of a Turbo Interleaver such as shown in FIG. 2 of size 1152 bits at Code Rate 1/3, using four (4) decoder iterations comparing Galois Field Interleavers to S-Random Interleavers, Random Interleavers for Bit Error Rate (BER) and for Frame Error Rate (FER).

[0015]   Corresponding reference characters indicate corresponding components throughout the several views of the drawings.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0016]   The following description of the presently contemplated best mode of practicing the invention is not to be taken in a limiting sense, but is made merely for the purpose of describing the general principles of the invention. The scope of the invention should be determined with reference to the claims.
[0017]   Referring to FIG. 1, a block diagram is shown of a digital communication system using Turbo Codes of a type that could be used to implement the teachings of the present invention. It comprises transmitter hardware including: a Transmitter Interface 102, an A/D Converter 104; a Segmentation Processor 106; a Turbo Coder 108; a Burst Formatter 110; a Modulator 112; a Transmitter (RF/IF) 114. It also comprises a Power Supply 124; a Timing and Control Processor 116; a Synthesizer and Oscillator 118; and a Switch 120. It comprises receiver hardware including: a Receiver Interface 134; a Turbo Decoder 132; an Equalizer 130; a Receiver/Demodulator 128; and a Receiver (RF/IF) Preamp Mixer 126.
[0018]   The transmitter receives an analog signal through a Transmitter Interface 102 and performs an A/D conversion at A/D Connector 104. The discrete samples generated from the A/D Converter 104 are fed to Segmentation Processor 106 where fixed-length data units of 44 octets are formed by fragmenting an initial MAC protocol data unit (IMPDU), and then the fixed length data units passed to the Turbo Coder 108 which uses an Interleaver to pseudo-randomize the input between 2 concatenated encoders and encodes the fixed length data units (data streams) and sends encoded data units (data streams) to the Burst Formatter 110.
[0019]   A burst, a series of repetitive waveforms at a prescribed time and amplitude lasting a short duration, is formed at Burst Formatter 110 and is passed to Modulator 112 where the burst is modulated by mixing with a carrier waveform of known frequency. Transmitter 114 transmits the modulated burst when the switch 120 connects antenna 122 to the Transmitter 114. The Synthesizer and Oscillator 118 keeps track of timing for the transmitter (RF/IF) 114 and for the

Timing and Control Processor 116 which controls when bursts are formatted.

[0020] When the Antenna 122 receives a burst and the receiver (RF/IF) Preamp Mixer 126 is connected to the antenna through the switch 120, the received burst is amplified by the receiver (RF/IF) Preamp Mixer 126, and then demodulated to remove the carrier waveform frequency. The Equalizer 130 filters the demodulated burst with filters adjusted so as to produce an enhanced digital signal which is next Turbo decoded by Turbo Decoder 132 through a concatenation of decoders and an Interleaver using feedback from each other decoder to decode information from the received burst. Decoded data is converted from digital to analog by D/A Converter 134 and passed through receiver interface 136 to another system for further processing as needed. Since the digital communication system of FIG. 1 would typically communicate using a variety of different information block sizes depending on the service requirements such as for voice or packet data, the embedded turbo code interleaver must be flexible enough to accommodate multiple block sizes without undue sacrifice in turbo code performance.

[0021] Referring to FIG. 2, a functional block diagram is shown of a representative turbo code encoder consisting of a parallel, concatenation of two simple constituent encoders (encoders) 10, 10', coupled to an Interleaver with memory (Interleaver) 16 and a puncturer 36. The first encoder 10 comprises: modular adders (or binary adders) 17, 20, 26, 28, 24, 25, and 30; shift register delay elements (or "shift registers") 18, 21, 22; a switch 12; output connections for an information bit X(t) and for parity bits $Y_0(t)$, $Y_1(t)$. The second encoder 10' comprises analogous hardware 17', 20', 26', 28', 24', 25', 30', 18', 21', 22', 12'. Output X(t) is coupled to the switch 12 coupled to input X(t). Output $Y_0(t)$ is coupled to modular adder 24 coupled to modular adder 20 at its output, which is coupled to register 18 and modular adder 17 at its input, which is coupled to switch 12. Output $Y_1(t)$ is coupled to modular adder 25 coupled to modular adder 28 at its output and to register 22 at its output. Modular adder 28 is coupled to modular adder 26 at its output and register 21 at its output; modular adder 26 is coupled to modular adder 17 at its output and to register 18 at its output. Modular adder 30 is coupled to modular adder 17 at its input. A detailed description of how the Turbo Coder of FIG. 2 operates in practice follows.

[0022] The two constituent encoders 10, 10' produce parity bits $Y_0(t)$, $Y_1(t)$ and $Y_0'(t)$, respectively, selected ones of which are removed from an output stream (output) of the two simple constituent encoders 10, 10' according to a prescribed puncturing pattern by the puncturer 36 in order to achieve a desired overall Turbo code rate. Both the first encoder (encoder #1) 10 and the second encoder (encoder #2) 10' process the same information bit stream and X(t) (or "information bits"), but the encoder #2 10' processes information bits X(t) in a different order than the order in which encoder #1 10 does since the Interleaver 16 processes the information bits X(t) before they reach encoder #2 10'. By rearranging an order of presentation of the information bits X(t), the Interleaver 16 serves to decorrelate the outputs of the two simple constituent encoders 10, 10' so that the information bits X(t) causing encoder #1 10 to produce a low-Hamming weight output are unlikely to cause encoder #2 10' to also produce a low-Hamming weight output.

[0023] In FIG. 2, the Interleaver 16 avoids mapping a "neighbor position" to a corresponding "neighbor position" of the interleaved bit sequence. The Interleaver 16 does this in a pseudo-random fashion by re-ordering bit locations in a random-looking predetermined fashion.

[0024] Both encoders 10, 10' produce, in addition to the information bits X(t) (or systematic bits), parity bits $Y_0(t)$ and $Y_1(t)$ which are punctured by puncturer 36 to achieve a desired overall Turbo Code rate.

[0025] Information bit stream X(t) is received at switch 12, and is processed in accordance with several modular adders of above and shift registers above which are hard-wired to represent two (2) numerator polynomials and one denominator polynomial.

[0026] Referring still to FIG. 2, a denominator polynomial d(D), representing Turbo Code "1010", is hardwired by the return feedback connection to modular adder 17 and its respective connections to modular adder 30. Before computing, three shift registers 18, 21, and 22 are first zeroed.

[0027] A first numerator polynomial over a denominator polynomial, representing Turbo Code "1101" is hardwired to return output $Y_0(t)$ by combining: X(t) with a result of modulator adder 17 to create a first bit W(t); the modular sum (second bit) of shift register 18 and W(t) from the modular adder 20; another zero bit (third bit) indicated by the lack of connection to the register 21; and the modular sum (fourth bit) of another register 22 and a result of modular adder 20 from modular adder 24.

The result is $Y_0(t)=W(t)+S_0(t)+S_2(t)$.

[0028] Information bit stream X(t) is presented in its original, uninterleaved order at a switch 12 and processed by the first encoder 10. In FIG. 2, the first encoder 10 is implemented as a linear feedback shift register whose transfer function is:

$$G(D) = \left[1 \quad \frac{1+D+D^3}{1+D^2+D^3} \quad \frac{1+D+D^2+D^3}{1+D^2+D^3} \right].$$

Thus, during an encoding step at time t≥0, a shift register contents of the shift register 18, 21, 22, are $S_0(t)$, $S_1(t)$, $S_2(t)$ and the information bit X(t) is present at the input to binary adder 17. The encoder 10 then produces its two coded output bits (coded bits) $Y_0(t)$, $Y_1(t)$ according to the following two summations:

$$Y_0(t)=W(t)+S_0(t)+S_2(t)$$

$$Y_1(t)=W(t)+S_0(t)+S_1(t)+S_2(t),$$

wherein

$$W(t)=X(t)+S_1(t)+S_2(t).$$

After the coded bits are output, the current encoding step at time t is completed by shifting the contents of the shift register 18, 21, 22 once to prepare for a next encoding step at time t+1. At time t+1:

[0029] $S_0(t+1)=W(t)$, $S_1(t+1)=S_0(t)$ and $S_2(t+1)=S_1(t)$. At the start of an encoding process at t=0, the shift register contents are initialized to zero, wherein $S_0(0)=S_1(O)=S_2(0)=0$. The second encoder 10' operates in the same fashion on an output of Interleaver 16 to produce another two (2) coded output bits.

[0030] Since the digital communication system of FIG. 1 would typically communicate using a variety of different information block sizes depending on the service requirements such as for voice or packet data, an embedded turbo code interleaver within turbo coder 108 must be flexible enough to accommodate multiple block sizes without undue sacrifice in turbo code performance. In its most general form, an interleaver design proposed herein consists of a collection of basic interleavers of various block lengths, an algorithm for selecting one of the basic interleavers to use as a "mother" interleaver, and a method of adapting the mother interleaver to produce a turbo interleaver of a particular desired length.

[0031] The basic interleavers are stored in a memory, which may be located within the Interleaver 16 as in FIG. 2, either as an explicit table of read or write indices or as a smaller set of parameters from which the table of read or write indices can be regenerated according to a predetermined algorithm.

[0032] A couple of simple examples will clarify these concepts. First, consider an interleaver of length 8 using a permutation n = (04261537). This permutation could be used as either a list of write (input sequence) addresses or read addresses. Let $d_I(0)$, $d_I(1)$, ..., $d_I(7)$ denote input data (input sequence) in their original sequence; and let $d_o(0)$, do(1), ..., $d_o(7)$ denote values of the same input data but in a permuted order. The interleaver could be implemented to write $d_I(0)$ to output position 0, $d_I(1)$ to output position 4, $d_I(2)$ to output position 2, $d_I(3)$ to output position 6, etc. In this case, the interleaver action can be expressed mathematically as

$$d_o(\pi(k))=d_I(k).$$

[0033] Alternately, the interleaver could be implemented to read data values from the input data according to the permutation n. That is, a first interleaved value $d_o(0)$ is read from input position 0, a second interleaved value $d_o(1)$ is read from input position 4, and so on. Mathematically,

$$d_o(k)=d_I(\pi(k)).$$

[0034] Neither interpretation is to be preferred; it is merely a matter of convention. For purposes of describing interleaver operations herein, the first interpretation (in which permutations specify write addresses for the interleaver) is used.

[0035] It should be noted, however, that in a turbo decoder, such as the turbo decoder 132 in FIG. 1, both interleaving and its inverse (de-interleaving) are used. If the interleaver is implemented to use the permutation n as write addresses, the de-interleaver can be implemented to use the permutation n as read addresses. This means that the interleaving and de-interleaving operations can share the same permutation generation hardware of software. It is not necessary to store descriptions of both the interleaver and its de-interleaver separately.

[0036] The permutation n = (04261537) arises from bit-reversal indexing. For example, input position 1 has a 3-bit binary representation 001 and is mapped to output position 4, which has 100 as its 3-bit binary representation. Likewise,

input position 3 (binary 011) is mapped to output position 6 (binary 110). In VLSI hardware or on some digital signal processing, which, optionally may be employed within the Interleaver 16 bit-reversed indexing is easily accomplished without special memory storage.

[0037]   The permutation n=(03614725) can be generated by the simple mathematical recursion:

$$\alpha=3; \quad \pi(0)=); \quad \pi(k) = \pi(k-1) + \alpha \pmod{8}.$$

Different recursions of this type can be described by the two parameters α and n(0). Thus, a family of basic interleavers based on simple recursive formulas could be represented by a small table of parameters stored in the memory. Advantageously, for an interleaver of large block length or a large set of interleavers of various block lengths, the ability to store a small table of parameters rather than the explicit permutations results in a large reduction in memory requirements. Thus, it is advantageous to design turbo interleavers in this way provided a parameterized family of interleavers results in good turbo code performance. These design issues are favorably addressed by the proposed invention.

[0038]   Given a family of basic interleavers of various block sizes represented and stored in memory in some fashion, a turbo device (either the turbo encoder 108 or the turbo decoder selects one of them for use in implementing an interleaver of specific length L. In one preferred embodiment of the invention, the lengths of the basic interleavers are all different, and the turbo device selects the basic interleaver having a smallest length N among all basic interleavers whose lengths are at least as big as the desired length L.

[0039]   In other embodiments, it may be desirable to have multiple basic interleavers all of the same length. For example, there may be an implementation advantage in having all basic interleavers have lengths that are integral powers of two. In such a design, there may be multiple basic interleavers of length $N=2^c$, each optimized for a different interval of block sizes between 2^{c-1} and 2^c. In such embodiments, the turbo device (turbo encoder 108, turbo decoder 132) first identifies a set of basic interleavers having a smallest length N among all the basic interleavers whose lengths are at least as big as the desired length L and then selects one of the basic interleavers in the set according to other selection criteria depending on L.

[0040]   Once a basic interleaver has been selected, it is then adapted to length L by the process referred to herein as pruning. Pruning refers to a discarding of permutation indices that are invalid for a pruned matrix. For example, one prunes the permutation n=(03614725) f length 8 on the integers modulo 8 to a new permutation of length 5 on the integers modulo 5 by ignoring the invalid indices 5, 6 and 7. Thus, a pruned permutation is Π *=(03142).

[0041]   The process of pruning, in accordance herewith, is further explained by an algorithm shown in FIG. 3. For simplicity, the algorithm assumes that the basic interleavers all have lengths that are integral multiples of two. The processing steps are as follows:

[0042]   The above rules are refined later herein so that Rule 1 and Rule 2 continues to be satisfied for Turbo Interleavers of any size N obtained from a single Interleaver of size 2^m by means of puncturing ($2^{m-1}<N\leq2^m$). Obtaining an Interleaver of any size N from a mother Interleaver of a larger size via puncturing is one aspect of this invention.

[0043]   A smaller Interleaver I_N, of size N_s is formed by using a pre-designed Interleaver matrix, I_2m, of size 2^m, where m is chosen such that it is the smallest integer for which $2^m \geq N_s$, i.e., the smallest power of two that is larger than or equal to the size N number of elements, N an integer, in the Interleaver I_N.

[0044]   A Smaller Interleaver, I_{N_s}, is then generated from the pre-designed Interleaver, I_2m by puncturing the predesigned Interleaver, I_2m.

[0045]   Thus, a Smaller Interleaver, I_{N_s}, is created by only accepting bit positions into the smaller Interleaver, I_{N_s}, from the original pre-designed Interleaver, I_2m, if the bit position value is smaller than the size of the smaller Interleaver, I_{N_s}, measured by the number of elements in the smaller Interleaver, N_s.

[0046]   This can be accomplished, for example, by a processor modified with a computer program, the steps of which are shown in FIG. 3, that initiates the following steps:

1) Initialize a counter i to zero, where i represents a new smaller Interleaver bit position, and j represents an original larger Interleaver bit position. This corresponds to Initialize Counter 310 in FIG. 3;

2) For every original bit position $I_2m[j]$; where j is from 0 to 2^m-1 (Check j 320 of FIG. 3), initiate the further steps:

[0047]   If $I_2m[j]<N_s$, set $I_{N_s}[i]=I_2m[j]$ and increment the counter i. These steps correspond to Check Larger I Element 330, Set Smaller I Element 340, and Increment Counter 350 respectively, of FIG. 3.

[0048]   Otherwise reject $I_2m[j]$ per Reject and Return 360 of FIG. 3. This program accepts, consecutively, from a first to a last bit position, any original Interleaver which has a value less than the smaller Interleaver size.

[0049]   Pruning is a key aspect of the invention described herein. It is the advantage that the method is easily imple-

mented in either a VLSI or a DSP and so provides an efficient mechanism for providing interleavers of arbitrary lengths without storing separate descriptions for every possible length. The set of basic interleavers are designed to be robust with respect to pruning in accordance with principles to be described in conjunction with a detailed, explicit design illustrative of the invention.

[0050]   The design of a turbo interleaver should take into account the structure of the constituent recursive convolutional codes in order to ensure that an overall Turbo code has a favorable Hamming weight spectrum ("weight spectrum") leading to good error correction performance. The weight spectrum of a linear binary code of length N is a tabulation giving a number of code words of each Hamming weight from O to N. A Hamming weight is a number of non-zero entries in the code word. Since the constituent code of a Turbo Code is recursive, it takes an input sequence of a Hamming weight of at least two (2) to cause a systematic encoder to leave an all zero state and later to return to the all zero state and therefore to generate a less desirable, low parity Hamming weight sequence. In general, a systematic, recursive encoder would generate a parity sequence of high Hamming weight for input sequences having a Hamming weight of one (1), since the encoded sequence upon leaving the all zero-state can never return to it. For recursive convolutional codes as constituent codes, the probability that both encoders generate encoded sequences that leave the all zero-state and later return to the all zero-state is the highest when the input sequence is of Hamming weight two (2).

[0051]   It is also observed that when the recursive eight-state constituent encoders have a primitive feedback polynomial of degree 3, an input sequence of Hamming weight two (2) can cause the first constituent encoder to generate a finite error event only if the two "1's" in the input sequence are separated by 6+7n (n is an integer) zeros. It is therefore important that any input sequence consisting of exactly two 1's separated by 6+7n (n∈N) zeros should not be mapped by the interleaver to a new sequence with two 1's now separated by 6+7m (m∈N) zeros. In that way the second encoder 10' will generate high parity Hamming weight when the first encoder generates low parity Hamming weight, and vice versa, corresponding to an input sequence of Hamming weight two (2).

[0052]   Even if the two 1's are separated by the undesirable 6+7n zeros in an input sequence of Hamming weight two, the corresponding parity Hamming weight will grow larger as n grows larger. Thus, it is less crucial to address the cases for n>1, as the most critical values for n are 0 followed by 1, respectively. This is because as n grows, parity Hamming weight grows sufficiently larger.

[0053]   The rules that are introduced in one embodiment of the invention to design Turbo Interleavers for eight-state Turbo codes are thus:

Rule 1: Minimize the occurrence of events:

$$|I[x]-I[x-7]|=7 \qquad\qquad (1)$$

wherein I[x] denotes the position that x is mapped to by the Interleaver matrix I.

Rule 2: If the first rule is satisfied with zero occurrences of equation (1), minimize the occurrence of event:

$$|I[x]-I[x-7]|=14, \ or$$
$$|I[x]-I[x-14]|=7, \ or \qquad\qquad (2)$$
$$|I[x]-I[x-14]|=14$$

[0054]   By following the above created rules, the probability of both of the encoders 10, 10' generating low-Hamming weight parity sequences is minimized.

[0055]   An explicit exemplary turbo interleaver design (exemplary design) will not be described in order to more fully illustrate and develop the concepts of the invention. In the exemplary design, each of the basic interleavers implemented by the Interleaver 16 is a two-dimensional block interleaver (or interleaver matrix) of dimension R x C, where R - 2^r is a number rows and C - 2^c is a number of columns. Conceptually, the input data (data) are written into the interleaver matrix row by row. Then row and column permutations are performed to randomize data positions. The data are then read out column by column. Specifically, given an input position $\ell = C \cdot i + j$, a corresponding output interleaved position will be given by mathematical formula $I(\ell)=R \cdot \Pi_i (j)+\rho(i)$, wherein Π_i is a column permutation applied to data in row i and wherein ρ is bit-reversed indexing, which is especially simple to implement and requires no additional parameter storage in the memory of the interleaver 16.

[0056]   One could, of course, make the ρ permutation different for different columns ad the expense of additional implementation complexity and increased storage requirements to specify each of the individual column permutations.

In either case, the ρ premutation(s) should perform pseudo-random interlacing of top and bottom halves of the interleaver matrix in order to facilitate on-the-fly implementation of pruning. Such interlacing ensures that, if $I(\ell)$ is an invalid index for a pruned interleaver, then $I(\ell+1)$ will be a valid index, assuming that no basic interleaver is pruned to half its length or beyond.

[0057] The proposed two-dimensional structure is advantageous for use in turbo interleaving for several reasons. Since the turbo interleaver is built in a structured way from simple constituent permutations that can be described by a small set of parameters, implementation complexity is small. When different constituent permutations are used from row to row, a composite interleaver permutation exhibits sufficient randomness to achieve good turbo code performance despite its low complexity. Furthermore, by choosing R and C appropriately, one can balance the "spreading capability" of the interleaver (how well is separates neighboring positions) and its "randomness" properties. The spreading capability of the interleaver is also important for the turbo interleaver in that it helps to enhance the overall "weight spectrum" of the turbo code. In general, spreading capability increases with increasing R, and randomness increases with increasing C.

[0058] Preferably, as a rule of thumb, for building interleaver matrixes in accordance with the invention, one would make R as large as possible without making C so small that the randomness produced by the permutations applied to each row is degraded.

[0059] Thus, in the illustrative designs presented below, the set of basic interleavers have the property that, in general, those of larger length use a larger number of rows R. This is an important aspect of the two-dimensional design.

[0060] In the illustrative designs, in accordance with one aspect of the invention presented below, the constituent permutations applied to rows of the interleaver matrix are based on a novel class of permutations derived from Galois field arithmetic.

[0061] A Galois Field (GF) with p^m elements is denoted as $GF(p^m)$, wherein p is a prime number and m is any integer greater than one (1). It can be formed from GF(p) using a primitive polynomial p(x) of degree m over GF(p)[x]. In the case of $GF(2^m)$, the roots of primitive polynomial p(x) of degree m over GF(2)[x], form a subset of the primitive elements in $GF(2^m)$. A primitive element in a Galois field with q elements has order q-1, i.e., the smallest positive integer n such that $\alpha^n = 1$ is n=q-1.

[0062] If $\propto$ is a primitive element in $GF(2^m)$, all of the other nonzero elements of $GF(2^m)$ can be obtained as consecutive powers of α.

$$GF(2^m) = \{0, \alpha^0 = 1, \alpha, \alpha^2, \alpha^3, ..., \alpha^{2m-2}\} \qquad (5)$$

[0063] Furthermore, every element of the field $GF(2^m)$ can be expressed in terms of $1, \alpha, \alpha^2, ..., \alpha^{m-1}$;

[0064] For example, GF(8) can be constructed from GF(2) using the primitive polynomial p(x)= x^3+x+1 over $GF(2)(x)$. Let α be a root of p(n). Multiplication of elements can be performed using the fact that α^7=1 (by definition since α is primitive in GF(8)). Addition of elements in GF(8) can be performed using equalities in terms of 1, α, and α^2 since $\alpha^3 = \alpha+1$ in Galois Field arithmetic (where $1 \equiv -1$).

[0065] An exemplary multiplication of elements in GF(8) is Equation (4).

$$\alpha^3 \bullet \alpha^6 = \alpha^9 = \alpha^2 \qquad (6)$$

[0066] An exemplary addition is Equation (5).

$$\alpha^3 + \alpha^4 = (\alpha+1) + (\alpha^2+\alpha) = \alpha^2 + 1 = \alpha^6 \qquad (7)$$

[0067] An Interleaver of size 2^m, $I_2{}^m$, is formed by the following four (4) steps:

(1) First, a matrix is filled row by row with bit positions starting with 0 in the upper leftmost position, and ending with (rxc-1, where 2^m=rxc as defined above) in the lower rightmost position. This is the conventional manner of filling Interleaver matrices.

[0068] Thus, an Interleaver matrix of size 32=4x8 would result in matrix (8):

$$\begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 \\ 24 & 25 & 26 & 27 & 28 & 29 & 30 & 31 \end{bmatrix} \qquad (8)$$

(2) Secondly, permute each row i (i=0,1,2,...,r-1) (within itself according to a predetermined rule. One method is to permute) according to the following permutation rules employing Galois Field arithmetic (9):

$$j \leftarrow \log_{\alpha}{}^{i_b}(\alpha^{i_o} + \alpha^j) \ \textit{for} \ j=0,1,2,3,\ldots,c-2$$
$$j \leftarrow \log_{\alpha}{}^{i_b}(\alpha^{i_o}) \qquad \textit{for} \ j=c-1 \qquad (9)$$

[0069] In Equation (9), ∞ is a root of the primitive polynomial p(n) used to construct GF(c), α^{ib} is primitive in GF(c) and i_0 is a designed integer between 0 and c-2 inclusive, and i_b is a predetermined integer, i_0 and i_b selected based upon certain design rules to be described.

[0070] Furthermore, by definition $\log_{\alpha} i_b(o)$ is set to (c-1) as a result of the second part of Equation (9).

[0071] An exemplary permutation would be such as is shown in Equation (10) for each row i (i=0,1,2,3), for an Interleaver of size 32 having 8 columns and 4 rows.

$$j \leftarrow \log_{\alpha}{}^{i_b}(\alpha^{i_o} + \alpha^j) \ \textit{for} \ j = 0, 1, 2, 3, 4, 5, 6$$
$$j \leftarrow \log_{\alpha}{}^{i_b}(\alpha^{i_o}) \qquad \textit{for} \ j = 7 \qquad (10)$$

[0072] For the sake of demonstration, Table 1 is constructed with constants i_b and i_0 to permute a Turbo Interleaver matrix of size N_s, within the set N, where $16 < N \leq 32$. The values of Table 1 are fabricated herein for the sake of the following example for the construction of I_{32}.

Table 1 - Constants

i	i_b	i_0
0	1	0
1	1	2
2	3	5
3	6	4

[0073] Thus, in accordance with Table 1, each row is shuffled such that:

$$\textit{row} \ i = 0: \ j \leftarrow \log_{\alpha}(\alpha^j + 1) \ \textit{for} \ j = 0, 1, 2, 3, 4, 5, 6$$
$$j \leftarrow \log_{\alpha}(1) \qquad \textit{for} \ j = 7 \qquad (11)$$

$$\text{row } i = 1: \quad j \leftarrow \log_\alpha(\alpha^j + \alpha^2) \text{ for } j = 0,1,2,3,4,5,6$$
$$j \leftarrow \log_\alpha(\alpha^2) \quad \text{for } j = 7 \tag{12}$$

$$\text{row } i = 2: \quad j \leftarrow \log_{\alpha^3}(\alpha^j + \alpha^5) \text{ for } j = 0,1,2,3,4,5,6$$
$$j \leftarrow \log_{\alpha^3}(\alpha^5) \quad \text{for } j = 7 \tag{13}$$

$$\text{row } i = 3: \quad j \leftarrow \log_{\alpha^6}(\alpha^j + \alpha^4) \text{ for } j = 0,1,2,3,4,5,6$$
$$j \leftarrow \log_{\alpha^6}(\alpha^4) \quad \text{for } j = 7 \tag{14}$$

[0074] The shuffling of each row results in a pseudo-random order of positions represented by sequences to the right of each of the arrows in equations (15), which represent the original bit positions which map to newly ordered sequences to the left of each arrow:

$$\begin{aligned}
&\text{row 0:} \quad (0,1,2,3,4,5,6,7) - (7,3,6,1,5,4,2,0) \\
&\text{row 1:} \quad (0,1,2,3,4,5,6,7) - (6,4,7,5,1,3,0,2) \\
&\text{row 2:} \quad (0,1,2,3,4,5,6,7) - (6,2,1,3,0,7,5,4) \\
&\text{row 3:} \quad (0,1,2,3,4,5,6,7) - (2,5,6,1,7,0,4,3)
\end{aligned} \tag{15}$$

[0075] The above shuffling results in an Interleaver matrix:

$$\begin{bmatrix}
7 & 3 & 6 & 1 & 5 & 4 & 2 & 0 \\
14 & 12 & 15 & 13 & 9 & 11 & 8 & 10 \\
22 & 18 & 17 & 19 & 16 & 23 & 21 & 20 \\
26 & 29 & 30 & 25 & 31 & 24 & 28 & 27
\end{bmatrix} \tag{16}$$

(3) Thirdly, each of the rows of Interleaver matrix I_{32} are shuffled or re-ordered according to any method that interlaces an upper half of the matrix resulting from the above permutations, with a lower half of the matrix.

[0076] One method of doing this is to re-order the rows according to a bit reversal on row index (e.g. as represented by the pattern (00,01,10,11) for a four-row matrix). From the above example matrix, this results in matrix:

$$\begin{bmatrix}
7 & 3 & 6 & 1 & 5 & 4 & 2 & 0 \\
22 & 18 & 17 & 19 & 16 & 23 & 21 & 20 \\
14 & 12 & 15 & 13 & 9 & 11 & 8 & 10 \\
26 & 29 & 30 & 25 & 31 & 24 & 28 & 27
\end{bmatrix} \tag{17}$$

(4) Fourthly, the contents of the resulting permuted and re-ordered matrix, or Interleaver matrix, are read out column by column to an encoder as in the case of a Block Interleaver.

[0077] In the above example, this results in the bit position sequence:

$$7\ 22\ 14\ 26\ 3\ 18\ 12\ 29\ 6\ 17\ 15\ 30\ 1\ 19\ 13\ 25\ 5\ 16\ 9\ 31\ 4\ 23\ 11 \qquad (18)$$
$$24\ 2\ 21\ 8\ 28\ 0\ 20\ 10\ 27$$

[0078] Permutations of the rows within themselves should be done in such a way that Rule 1 and Rule 2 are satisfied for any Interleaver size N obtained from an original Interleaver of size 2^m where $2^{m-1}<N\leq 2^m$.

[0079] The preferred basic interleaver structure, described previously herein allows the formulation of simple design criteria that help ensure robustness to pruning. The key observation is that, because of the way Interleavers of size 2^m are constructed, for any window of size 2W, W an integer, there is at most W indices that must be pruned in order to obtain an Interleaver of size N wherein $2^{m-1}<N\leq 2^m$. An Interleaver of 17 elements is obtained by pruning the interleaver of equation (18).

$$7\ 14\ 3\ 12\ 6\ 15\ 1\ 13\ 5\ 16\ 9\ 4\ 11\ 2\ 8\ 0\ 10. \qquad (19)$$

resulting in an Interleaver matrix having elements of 7 14 3 12 6 15 1 13 5 16 9 4 11 2 8 0 10.

[0080] Thus, the previously discussed rules for good turbo interleaver design - Rules 1 and 2 given in equations (1) and (2) - can be generalized to provide rules for the design of good turbo interleavers that are robust to pruning. The modified rules are as follows:

Modified Rule 1: Minimize the occurrence of events.

$$|I[x]-I[x-j]|=7 \quad where \quad 7\leq j\leq 14,\ j\in N \qquad (20)$$

Modified Rule 2: If the first modified rule is satisfied with zero occurrence, minimize the occurrence of events.

$$|I[x]-I[x-j]|=7,\ or \qquad (21)$$

$$|I[x]-I[x-j]|=14 \quad where \quad 7\leq j\leq 28,\ j\in N \qquad (22)$$

[0081] A third Rule is also introduced: Rule 3: If the Modified Rule 1 and Modified Rule 2 are satisfied with zero occurrence, maximize the variable S such that, neighbor positions within a window size S are not mapped to neighbor positions within a window of size S.

[0082] Because of the key observation stated before the modified rules, it can be seen that if an Interleaver of size 2^m satisfies Modified Rule 1 and Modified Rule 2, then all of the Interleavers of size N ($2^{m-1}<N\leq 2^m$) obtained by pruning satisfy Rule 1 and Rule 2 stated at the beginning of this invention.

[0083] In designing preferred integer constants i_o and i_b for the Galois Field permutations to achieve pseudo-randomness in a Galois Field Interleaver, performance of constructed Interleaver matrices are measured according to how well they meet the modified rules above.

[0084] Interleaver matrices of size 128, 256, 512, 1024, 2048 and 4096 are constructed in accordance with the modified rules to yield near optimal Interleaver matrices of any size N where $64 < N \leq 4096$.

[0085] The primitive polynomials used to construct GF(c), where c is the number of columns of the Interleaver matrix are as follows:

$$
\begin{aligned}
p(x) &= x^4 + x + 1 & to \quad construct \quad & GF(16) \\
p(x) &= x^5 + x^2 + 1 & to \quad construct \quad & GF(32) \\
p(x) &= x^6 + x + 1 & to \quad construct \quad & GF(64) \\
p(x) &= x^7 + x^3 + 1 & to \quad construct \quad & GF(128) \\
p(x) &= x^8 + x^4 + x^3 + x^2 + x & to \quad construct \quad & GF(256) \\
p(x) &= x^9 + x^4 + 1 & to \quad construct \quad & GF(512)
\end{aligned}
\qquad (23)
$$

[0086] Table 2 shows best values of i_b and i_o, as defined and determined above, for each row index i for each Interleaver matrix size specified.

Table 2: Galois Field (GF) Turbo Interleavers of Size $2^m=rxc$

Row Index i	Int. Size: 128=8x16		Int. Size: 256=8x32		Int. Size: 522=16x32		Int. Size: 1024=16x64		Int. Size: 2048=32x64		Int. Size: 4096=32x128	
	i_b	i_0	i_b	i_0	i_b	i_0	i_b	i_0	i_b	i_0	i_b	i_0
0	1	13	22	18	3	11	19	26	50	29	11	11
1	8	2	8	5	28	8	28	17	20	55	20	89
2	6	14	4	1	22	18	60	26	13	32	82	13
3	14	8	15	12	27	2	6	34	15	38	120	33
4	12	13	13	24	11	5	60	25	51	17	105	2
5	5	4	25	23	12	8	38	58	37	20	69	71
6	6	13	27	16	14	9	5	41	50	34	23	73
7	9	14	30	16	7	8	40	25	24	4	87	70
8					9	24	55	32	36	32	72	64
9					16	14	31	26	0	8	73	95
10					6	28	16	38	25	31	36	14
11					17	11	28	50	37	61	102	108
12					2	9	46	23	38	62	64	21
13					24	3	40	22	27	25	109	67
14					3	2	32	55	41	10	42	14
15					14	11	21	19	51	43	27	106
16									5	37	64	63
17									43	10	13	17
18									54	41	5	65
19									4	26	46	62
20									44	10	111	116
21									19	40	68	12
22									26	17	48	65
23									60	44	3	53
24									23	16	60	66

(continued)

Row Index i	Int. Size: 128=8x16 $i_b i_0$		Int. Size: 256=8x32 $i_b i_0$		Int. Size: 522=16x32 $i_b i_0$		Int. Size: 1024=16x64 i_0	Int. Size: 2048=32x64 $i_b i_0$		Int. Size: 4096=32x128 $i_b i_0$	
25								19	39	47	90
26								38	58	126	59
27								47	54	115	1
28								13	38	113	38
29								46	7	12	9
30								46	22	57	75
31								17	13	55	6

[0087] In all cases of Interleavers and associated matrices designed from the above constants in accordance with the invention, Modified Rule 1 and Modified Rule 2 is completely satisfied except for a small number of

[0088] In one embodiment, a computer search determines the constants such that the Modified Rules are satisfied.

[0089] Referring to FIG. 4, simulation results are shown for a random, S-random and the new Galois Field Interleaver in accordance with the present invention of size 1024 over AWGN channel with overall Turbo code rate 1/2, wherein the encoder consists of eight-state constituent encoders with the transfer function:

$$G(D) = \left[1, \frac{1+D+D^3}{1+D^2+D^3} \right] \qquad (24)$$

[0090] Curve 410 of FIG. 4 shows that with four (4) decoder iterations, the Galois Field (GF) Interleaver has about a 0.1 dB gain with respect to a comparable S-Random Interleaver (S=12) at a bit error rate of 10^{-5}. A comparable random Interleaver has the worst performance. For Frame Error Rate performance, curve 420 shows that a (GF) Interleaver is also the best performing Interleaver.

[0091] FIG. 5 illustrates the corresponding performance for eight (8) decoder iterations. Curve 510 illustrates the Bit Error Rate performance compared to the others. Curve 520 illustrates the Frame Error Rate performance compared to the others. Results are consistent with FIG. 4.

[0092] FIG. 6 illustrates the performance of Turbo Interleavers of size 1152 over AWGN channel with overall Turbo code rate 1/3, and with four (4) decoder iterations. The GF Interleaver of size 1152 is formed from a (GF) Interleaver of size 2048 in accordance with this present invention. In this case, performance curves 310 for Bit Error Rate and 320 for Frame Error Rate illustrate that (GF) Interleavers have comparable performance with S-random Interleaver (S=13).

[0093] The illustrative design of Table 2 can be simplified by further restricting the choice of parameters. For example, the hardware implementation as well as storage requirements are reduced if the parameter i_b is made constant and equal to 1. In this case, the parameters describing the constituent permutations to be applied within each row R should be re-optimized with respect to Modified Rules 1 and 2.

[0094] Table 4 shows the near optimal values of integer constant i_0 for each row index i and each Interleaver matrix size, if i_b is 1.

Table 4: Simplified Galios Field (GF) Turbo Interleavers of Size 2^m=rxc

Row Index i	Int. Size: 256= 8x32 i_0	Int. Size: 512= 16x32 i_0	Int. Size: 10x24= 16x64 i_0	Int. Size: 2048 32x64 i_0	Int. Size: 4096= 32x128 i_0	Int. Size: 8192= 64x256 i_0	Int. Size: 16384= 64x256 i_0	Int. Size: 32768= 64x512 i_0
0	1	20	44	54	33	42	209	282
1	4	1	40	52	84	89	35	107
2	22	14	3	38	56	3	91	226
3	3	17	10	42	110	17	68	132
4	5	5	8	5	92	32	242	391
5	29	21	45	13	19	24	252	362
6	28	24	7	45	62	23	131	119
7	9	28	28	60	50	11	208	139
8		19	37	25	45	109	29	129
9		12	2	2	73	104	175	446
10		30	17	27	14	86	37	65
11		16	14	59	104	54	233	207
12		29	55	53	78	60	12	95
13		2	48	29	103	52	141	153

(continued)

Row Index i	Int. Size: 256= 8x32 i_0	Int. Size: 512= 16x32 i_0	Int. Size: 10x24= 16x64 i_0	Int. Size: 2048 32x64 i_0	Int. Size: 4096= 32x128 i_0	Int. Size: 8192= 64x256 i_0	Int. Size: 16384= 64x256 i_0	Int. Size: 32768= 64x512 i_0
14		25	19	33	98	55	196	208
15		23	12	20	59	95	239	399
16				61	67	102	160	20
17				4	46	51	150	51
18				57	0	72	20	8
19				1	74	27	62	77
20				30	38	107	240	385
21				58	36	33	220	422
22				35	124	110	42	434
23				40	61	29	235	509
24				7	48	28	21	168
25				3	112	94	58	273
26				6	111	105	10	81
27				41	87	5	119	w465
28				18	49	63	115	219
29				28	125	16	61	319
30				32	44	64	176	177
31				48	93	81	228	140
32						41	107	60
33						112	14	288
34						96	87	68
35						100	125	80
36						124	24	183
37						4	254	293
38						7	179	121
39						45	127	136
40						10	33	96
41						74	149	186
42						111	226	269
43						84	36	150
44						20	80	335
45						75	109	138
46						26	11	41
47						117	133	144
48						93	210	202
49						103	117	218

(continued)

Row Index i	Int. Size: 256= 8x32 i_0	Int. Size: 512= 16x32 i_0	Int. Size: 10x24= 16x64 i_0	Int. Size: 2048 32x64 i_0	Int. Size: 4096= 32x128 i_0	Int. Size: 8192= 64x256 i_0	Int. Size: 16384= 64x256 i_0	Int. Size: 32768= 64x512 i_0
50						0	30	357
51						66	40	238
52						78	138	22
53						92	79	299
54						37	16	297
55						91	216	468
56						71	198	24
57						40	143	161
58						43	248	328
59						38	69	237
60								
61						87	104	14
62						50	203	

[0095] While the invention herein disclosed has been described by means of specific embodiments and applications thereof, numerous modifications and variations could be made thereto by those skilled in the art without departing from the scope of the invention set forth in the claims.

[0096] For example, in the preferred embodiments, the constituent permutations applied within each row of the interleaver matrix were based on discrete logarithms in a Galois field. It is clear that this is only a particular example of a broad class of permutations based on Galois field arithmetic, which may be employed in accordance herewith. Optimally, another closely related choice would be to take a non-primitive element $\beta \varepsilon GF(C)$ of multiplicative order ord(β) and define

[0097] $\Pi_i(J) = \log(\beta^{io} + \beta^j)$, (j = 0, 1, 2, ..., ord(β)-1) to produce a constituent permutation of length ord (β)-1. Alternatively, a logarithm of a different linear or affine function of β^j may be employed. More generally, one could take permutation mapping data at position i=0, 1, 2, ..., ord(β)-1 to a new position

$$\pi_i(j) = f(\beta^j),$$

wherein f is any integer-valued function acting on finite field GF(C) and β is a non-zero element in GF(C) of multiplicative order ord(β).

[0098] In yet another alternative embodiment, the finite field(s) need not be binary-that is C need not be a power of 2.

Claims

1. A method of turbo-interleaving blocks of indexed data, interleaving being used in the construction of a turbo code, the blocks of indexed data having indices and the blocks having a varying length, the method including the steps of:

   providing a set of basic interleavers having a matrix structure including rows and columns, each basic interleaver using a family of one or more row permutations employing Galois field arithmetic of the indexed data based on consecutive powers of a primitive element, each row permutation permuting a row within itself and each basic interleaver having a fixed length 2^m,
   selecting one of the basic interleavers based upon a desired interleaver length N such that its length meets the relationship $2^{m-1} < N \leq 2^m$; and
   pruning the selected basic interleaver to produce an interleaver having the desired interleaver length N, wherein

m is the smallest integer which power of two is larger than or equal to the size N.

2. The method of Claim 1 wherein the step of pruning the selected basic interleaver includes deleting indexed data having indices higher than required for a permutation of length N.

3. The method of Claim 2 wherein the family of basic interleavers consists of "two-dimensional permutations" and further comprising the steps of:

computing the "two-dimensional permutations", including the steps of:

(a) writing the indexed data into an interleaver matrix having one or more rows R in each of two dimensions;
(b) permuting the indexed data in the one or more rows in at least one of the two dimensions, to produce constituent permutations, said constituent permutations possibly being different from one row to another row in each of the two dimensions; and
(c) reading out the indexed data from the interleaver matrix.

4. The method of Claim 3 further including the steps of:

providing an interleaver device (16) for the interleaving blocks of indexed data, the interleaver device (16) further comprising a memory device for storing descriptions of the basic interleavers; and
storing the descriptions of the basic interleavers in the memory device.

5. The method of Claim 4 wherein the constituent permutations are pseudo-random permutations described by mathematical operations involving a limited number of parameters, wherein an amount of storage required for storing the limited number of parameters is less than another amount of storage required for storing a vector representation of a totality of permuted indices of the constituent permutations.

6. The method of Claim 5 wherein at least one of the constituent permutations is a Galois field permutation that maps indexed data at positions i=1, 2,...ord(α)-1 to a new position $I(i)= f(\alpha^i)$ in the at least one of the two dimensions, wherein f is an integer-valued function acting on the finite field GF(q) and wherein α is a non-zero element in the finite Galois field GF(q) of multiplicative order ord(α); and applying at least one Galois permutation to the indexed data along the at least one of the two dimensions.

7. The method of Claim 5 further including the steps of:

providing a turbo code encoder comprising two or more constituent code encoders connected together by the interleaver device; and
turbo encoding the indexed data after the interleaving by the interleaver device of the turbo code encoder.

8. The method of Claim 5 wherein the lengths of the constituent permutations, applied along the at least one of the two dimensions are even integers.

9. The method of Claim 8 wherein the constituent permutations with even lengths serve to interlace the indexed data from a lower half and an upper half of the interleaver matrix, in the at least one of the two dimensions to which the constituent permutations with even lengths is applied, and in a consistent manner.

10. The method of Claim 9 wherein the lengths of the constituent permutations, applied along at least one of the two dimensions, are integral powers of two.

11. The method of Claim 10 wherein at least one of the constituent permutations with length and integral power of 2 maps indexed data at a position i, corresponding to index i, to a new position $I(i)$, corresponding to a position defined by the function $I(i)$, wherein I(i) is an integer whose binary representation is a reverse of the binary representation of i.

12. The method of Claim 1, further including the step of:

providing a turbo encoder consisting of two or more constituent encoders connected together, wherein the interleaving is performed by the turbo encoder.

13. A system for turbo-interleaving blocks of indexed data, having indices, the blocks having a varying length, the system comprising:

a parallel concatenation of two or more constituent encoders for recursive convolutional codes having a recursion period p; and
an interleaver device (16) coupled to the parallel concatenation of the two or more constituent encoders for performing the steps of:

accessing stored descriptions of basic interleavers, the basic interleavers having a matrix structure including rows and columns, each basic interleaver using a family of one or more row permutations employing Galois field arithmetic of the indexed data based on consecutive powers of a primitive element, each row permutation permuting a row within itself and each basic interleaver having a fixed length 2^m;
identifying a group of the basic interleavers having a length greater than or equal to a desired interleaver length N;
selecting one of the basic inteleavers having a length 2^m which is smallest among the group of the basic interleaves so that; $2^{m-1} < N \leq 2^m$; and
pruning the selected one of the basic interleavers to produce an
interleaver having the desired interleaver length N,
wherein m is the smalles integer which power of two is larger than or equal to the size N

Patentansprüche

1. Verfahren zur Turbo-Verschachtelung von Blöcken indizierter Daten, wobei eine Verschachtelung in der Konstruktion eines Turbo-Kodes verwendet wird, die Blöcke indizierter Daten Indizes haben und die Blöcke eine unterschiedliche Länge aufweisen, wobei das Verfahren folgende Schritte enthält:

Vorsehen eines Satzes von Basisverschachtelern mit einer Matrixstruktur, die Reihen und Spalten enthält, wobei jeder Basisverschachteler eine Familie einer oder mehrerer Reihenpermutation(en) verwendet, unter Nutzung einer Galois-Feldarithmetik der indizierten Daten auf Basis konsekutiver Potenzen eines primitiven Elements,
wobei jede Reihenpermutation eine Reihe in sich selbst permutiert und jeder Basisverschachteler eine festgesetzte Länge 2^m hat,
Auswählen eines der Basisverschachtelers auf Grundlage einer gewünschten Verschachtelerlänge N,
so dass seine Länge das Verhältnis $2^{m-1}<N\leq 2^m$ erfüllt; und
Kürzen des gewählten Basisverschachtelers zur Erzeugung eines Verschachtelers mit der gewünschten Verschachtelerlänge N, wobei m die kleinste ganze Zahl ist, deren Potenz von zwei größer oder gleich der Größe N ist.

2. Verfahren nach Anspruch 1, wobei der Schritt zum Kürzen des gewählten Basisverschachtelers das Löschen indizierter Daten mit Indizes, die höher als für eine Permutation der Länge N erforderlich sind, enthält.

3. Verfahren nach Anspruch 2, wobei die Familie von Basisverschachtelern aus "zweidimensionalen Permutationen" besteht und des Weiteren umfassend die folgenden Schritte:

Berechnen der "zweidimensionalen Permutationen",
einschließlich der Schritte:

(a) Schreiben der indizierten Daten in eine Verschachtelermatrix mit einer oder mehreren Reihe(n) R in jeder der zwei Dimensionen;
(b) Permutieren der indizierten Daten in der einen oder den mehreren Reihe(n) in mindestens eine der zwei Dimensionen, um Konstituentenpermutationen zu erzeugen, wobei sich die Konstituentenpermutationen möglicherweise von einer Reihe zur nächsten in jeder der zwei Dimensionen unterscheiden; und
(c) Auslesen der indizierten Daten aus der Verschachtelermatrix.

4. Verfahren nach Anspruch 3, des Weiteren enthaltend die folgenden Schritte:

Bereitstellen einer Verschachtelungsvorrichtung (16) zur Verschachtelung von Blöcken indizierter Daten, wobei

die Verschachtelungsvorrichtung (16) des Weiteren eine Speichervorrichtung zum Speichern von Beschreibungen der Basisverschachteler aufweist; und

Speichern der Beschreibungen der Basisverschachteler in der Speichervorrichtung.

5. Verfahren nach Anspruch 4, wobei die Konstituentenpermutationen pseudo-zufällige Permutationen sind, die durch mathematische Operationen beschrieben sind, die eine begrenzte Anzahl von Parametern enthalten, wobei eine Speichermenge, die zum Speichern der begrenzten Anzahl von Parametern erforderlich ist, geringer ist als eine andere Speichermenge, die zum Speichern einer Vektordarstellung einer Gesamtheit permutierter Indizes der Konstituentenpermutationen erforderlich ist.

6. Verfahren nach Anspruch 5, wobei mindestens eine der Konstituentenpermutationen eine Galois-Feldpermutation ist, die indizierte Daten an Positionen 1=1, 2,... ord(α)-1 an einer neuen Position I(i)=f(α') in der mindestens einen der zwei Dimensionen abbildet, wobei f eine ganzzahlige Funktion ist, die auf das finite Feld GF(q) wirkt, und wobei α ein Element ungleich Null in dem finiten Galois-Feld GF(q) einer multiplikativen Ordnung ord(α) ist; und

Anwenden mindestens einer Galois-Permutation an den indizierten Daten entlang der mindestens einen der zwei Dimensionen.

7. Verfahren nach Anspruch 5, enthaltend die folgenden Schritte:

Bereitstellen eines Turbo-Kode-Kodierers,
aufweisend zwei oder mehr Konstituenten-Kode-Kodierer, die durch die Verschachtelungsvorrichtung miteinander verbunden sind; und
Turbo-Kodieren der indizierten Daten nach der Verschachtelung durch die Verschachtelungsvorrichtung des Turbo-Kode-Kodierers.

8. Verfahren nach Anspruch 5, wobei die Längen der Konstituentenpermutationen, die entlang der mindestens einen der zwei Dimensionen angewendet werden, gerade ganze Zahlen sind.

9. Verfahren nach Anspruch 8, wobei die Konstituentenpermutationen mit geraden Längen zum Verschachteln der indizierten Daten von einer unteren Hälfte und einer oberen Hälfte der Verschachtelermatrix in der mindestens einen der zwei Dimensionen, an welcher die Konstituentenpermutationen mit geraden Längen angewendet werden, und auf beständige Weise dienen.

10. Verfahren nach Anspruch 9, wobei die Längen der Konstituentenpermutationen, die entlang der mindestens einen der zwei Dimensionen angewendet werden, Integralpotenzen von zwei sind.

11. Verfahren nach Anspruch 10, wobei mindestens eine der Konstituentenpermutationen mit Länge und Integralpotenz von 2 indizierte Daten an einer Position 1, entsprechend Index 1, an einer neuen Position I(i) entsprechend einer Position, die durch die Funktion I(i) definiert ist, abbildet, wobei I(i) eine ganze Zahl ist, deren binäre Darstellung eine Umkehr der binären Darstellung von i ist.

12. Verfahren nach Anspruch 1, des Weiteren enthalten den folgenden Schritt:

Bereitstellen eines Turbo-Kodierers, bestehend aus zwei oder mehr Konstituentenkodierern, die miteinander verbunden sind, wobei die Verschachtelung vom Turbo-Kodierer ausgeführt wird.

13. System zur Turbo-Verschachtelung von Blöcken indizierter Daten, die Indizes haben, wobei die Blöcke eine unterschiedliche Länge aufweisen, wobei das System aufweist:

eine Parallelverkettung von zwei oder mehr Konstituentenkodierern für rekursive Faltungskodes mit einer Rekursionsperiode p; und
eine Verschachtelungsvorrichtung (16), die an die Parallelverkettung der zwei oder mehr Konstituentenkodierer gekoppelt ist, um folgende Schritte auszuführen:

Zugreifen auf gespeicherte Beschreibungen von Basisverschachtelern, wobei die Basisverschachteler eine Matrixstruktur haben, die Reihen und Spalten enthält, wobei jeder Basisverschachteler eine Familie einer oder
mehrerer Reihenpermutation(en) verwendet, unter Nutzung einer Galois-Feldarithmetik der indizierten Da-

ten auf Basis konsekutiver Potenzen eines primitiven Elements, wobei jede Reihenpermutation eine Reihe in sich selbst permutiert und jeder Basisverschachteler eine festgesetzte Länge 2^m hat;

Identifizieren einer Gruppe der Basisverschachteler mit einer Länge größer oder gleich einer gewünschten Verschachtelerlänge N,

Auswählen eines der Basisverschachteler mit einer Länge 2^m, die die geringste in der Gruppe von Basisverschachtelern ist, so dass $2^{m-1} < N \leq 2^m$; und

Kürzen des einen gewählten Basisverschachtelers zur Erzeugung eines Verschachtelers mit der gewünschten Verschachtelerlänge N,

wobei m die kleinste ganze Zahl ist, deren Potenz von zwei größer oder gleich der Größe N ist.

Revendications

1. Procédé de blocs d'entrelacement turbo de données indexées, l'entrelacement étant utilisé dans la construction d'un turbo code, les blocs de données indexées ayant des indices et les blocs ayant une longueur variable, le procédé comprenant les étapes suivantes :

   fournir un ensemble d'entrelaceurs de base ayant une structure de matrice comprenant des rangées et des colonnes, chaque entrelaceur de base utilisant une famille d'une ou plusieurs permutations de rangées utilisant une arithmétique à champs de Galois des données indexées basées sur des puissances consécutives d'un élément primitif, chaque permutation de rangée permutant une rangée à l'intérieur d'elle-même et chaque entrelaceur de base ayant une longueur fixée à 2^m,

   sélectionner un des entrelaceurs de base sur la base d'une longueur souhaitée N d'entrelaceur de manière à ce que sa longueur satisfasse la relation $2^{m1} < N \leq 2^m$; et

   élaguer l'entrelaceur de base sélectionné pour produire un entrelaceur ayant la longueur d'entrelaceur souhaitée N, où m est l'entier le plus petit dont la puissance de deux est supérieure ou égale à la taille N.

2. Procédé selon la revendication 1, dans lequel l'étape comprenant d'élaguer l'entrelaceur de base sélectionné comprend de supprimer des données indexées ayant des indices plus élevés que nécessaire pour une permutation de longueur N.

3. Procédé selon la revendication 2, dans lequel la famille d'entrelaceurs de base consiste en des « permutations à deux dimensions », et comprenant en outre les étapes suivantes :

   calculer les « permutations à deux dimensions », y compris les étapes suivantes :

   (a) écrire les données indexées dans une matrice d'entrelaceur ayant une ou plusieurs rangées R dans chacune des deux dimensions ;
   (b) permuter les données indexées dans une ou plusieurs rangées dans au moins une des deux dimensions, pour produire des permutations de composants, lesdites permutations de composants pouvant être différentes d'une rangée à l'autre dans chacune des deux dimensions ; et
   (c) lire les données indexées depuis la matrice d'entrelaceur.

4. Procédé selon la revendication 3, comprenant en outre les étapes suivantes :

   fournir un dispositif d'entrelaceur (16) pour les blocs d'entrelacement de données indexées, le dispositif d'entrelaceur (16) comprenant en outre un dispositif de mémoire pour stocker des descriptions des entrelaceurs de base ; et
   stocker les descriptions des entrelaceurs de base dans le dispositif de mémoire.

5. Procédé selon la revendication 4, dans lequel les permutations de composants sont des permutations pseudo-aléatoires décrites par des opérations mathématiques impliquant un nombre limité de paramètres, où une quantité de stockage requise pour stocker le nombre limité de paramètres est inférieure à une autre quantité de stockage requise pour stocker une représentation vectorielle d'une totalité d'indices permutés des permutations de composants.

6. Procédé selon la revendication 5, dans lequel au moins une des permutations de composants est une permutation à champs de Galois qui mappe des données indexées aux positions $1=1, 2, ... \text{ord}(\alpha)-1$ à une nouvelle position

$l(i)=f(\alpha')$ dans l'au moins une des deux dimensions, où f est une fonction à valeur entière agissant sur le champs fini GF(q) et où α est un élément non nul dans le champs de Galois fini GF(q) d'ordre multiplicatif ord (α) ; et appliquer au moins une permutation de Galois aux données indexées suivant l'au moins une des deux dimensions.

7. Procédé selon la revendication 5, comprenant en outre les étapes suivantes :

    fournir un codeur de turbo code comprenant deux, ou plus, codeurs de code de composants connectés ensemble par un dispositif d'entrelaceur ; et
    turbo-coder les données indexées après l'entrelacement par le dispositif d'entrelaceur du codeur de turbo code.

8. Procédé selon la revendication 5, dans lequel les longueurs des permutations de composants, appliquées suivant l'au moins une des deux dimensions, sont des entiers pairs.

9. Procédé selon la revendication 8, dans lequel les permutations de composants ayant des longueurs paires servent à entrelacer les données indexées depuis une moitié inférieure et une moitié supérieure de la matrice d'entrelaceur, dans l'au moins une des deux dimensions à laquelle les permutations de composants ayant des longueurs paires s'appliquent, et de manière cohérente.

10. Procédé selon la revendication 9, dans lequel les longueurs des permutations de composants, appliquées suivant au moins une des deux dimensions, sont des puissances entières de deux.

11. Procédé selon la revendication 10, dans lequel au moins une des permutations de composants ayant une longueur et une puissance entière de 2 mappe des données indexées à une position i, correspondant à l'index i, à une nouvelle position $l(i)$, correspondant à une position définie par la fonction $l(i)$, où l(i) est un entier dont la représentation binaire est un inverse de la représentation binaire de i.

12. Procédé selon la revendication 1, comprenant en outre l'étape suivante :

    fournir un turbo codeur se constituant de deux, ou plus, codeurs de composants connectés ensemble, où l'entrelacement est exécuté par le turbo codeur.

13. Système pour effectuer un entrelacement turbo de blocs de données indexées, ayant des indices, les blocs ayant une longueur variable, le système comprenant :

    une concaténation parallèle de deux, ou plus, codeurs de composants pour des codes de convolution récursive ayant une période de récursion p ; et
    un dispositif d'entrelaceur (16) couplé à la concaténation parallèle des deux, ou plus, codeurs de composants pour exécuter les étapes suivantes :

        accéder à des descriptions stockées d'entrelaceurs de base, les entrelaceurs de base ayant une structure de matrice comprenant des rangées et des colonnes, chaque entrelaceur de base utilisant une famille d'une ou plusieurs permutations de rangées utilisant une arithmétique à champs de Galois des données indexées basées sur des puissances consécutives d'un élément primitif, chaque permutation de rangée permutant une rangée à l'intérieur d'elle-même et chaque entrelaceur de base ayant une longueur fixée à 2^m;
        identifier un groupe des entrelaceurs de base ayant une longueur supérieure ou égale à une longueur d'entrelaceur souhaitée N ;
        sélectionner un parmi les entrelaceurs de base ayant une longueur de 2^m qui est la plus petite parmi le groupe des entrelaceurs de base de manière à ce que $2^{m-1} < N \le 2^m$; et
        élaguer l'entrelaceur de base sélectionné parmi les entrelaceurs de base pour produire un entrelaceur ayant la longueur d'entrelaceur souhaitée N,
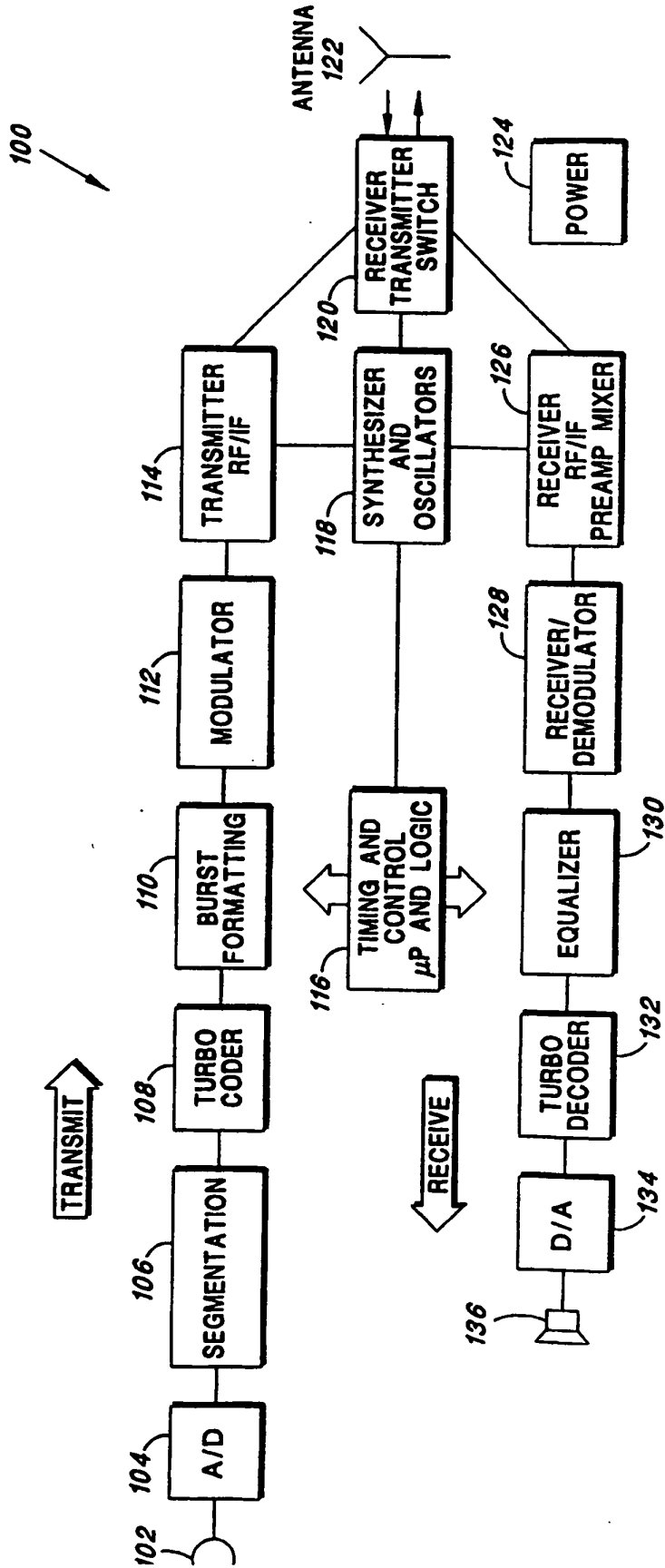   où m est le plus petit entier dont la puissance de deux est supérieure ou égale à la taille N.

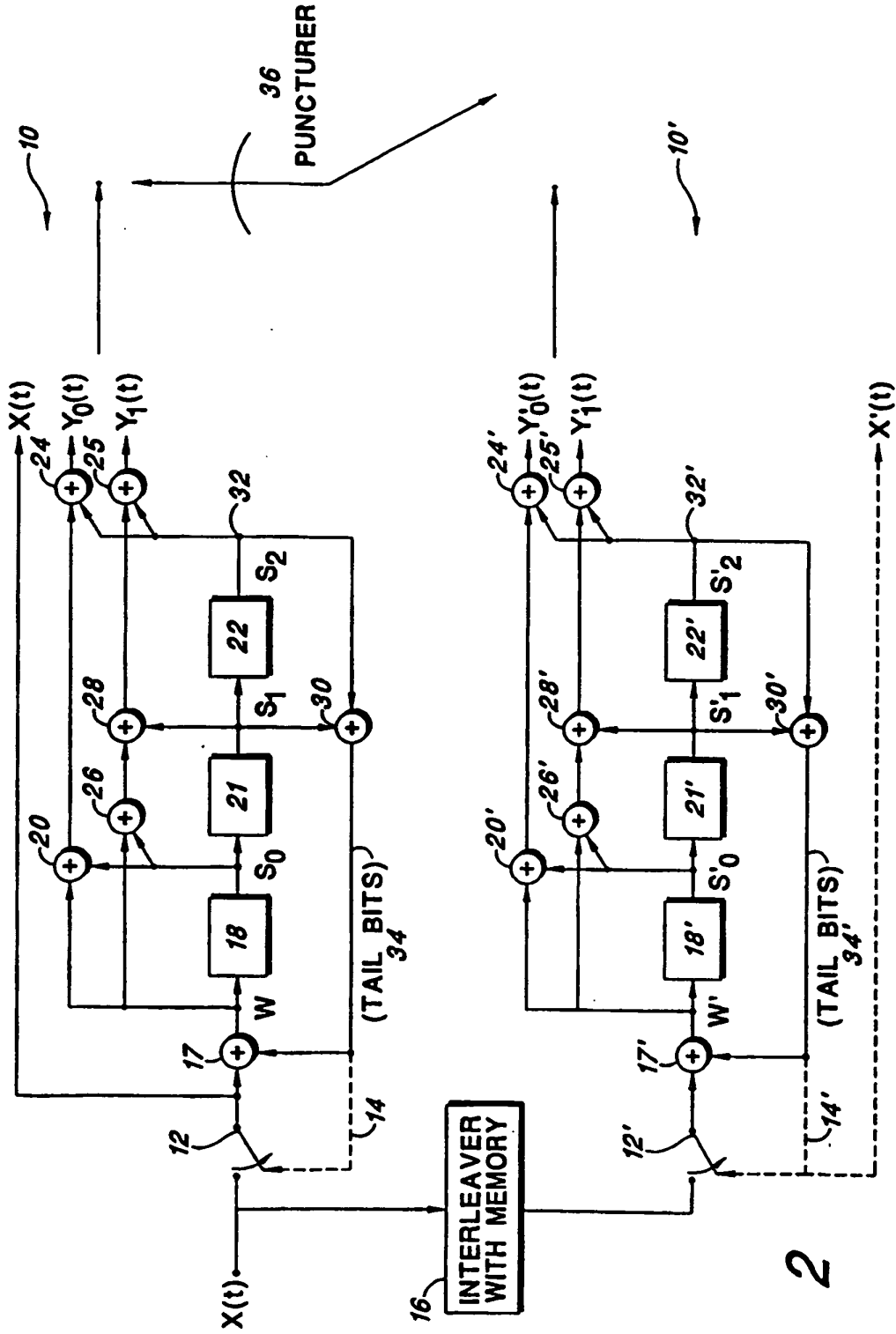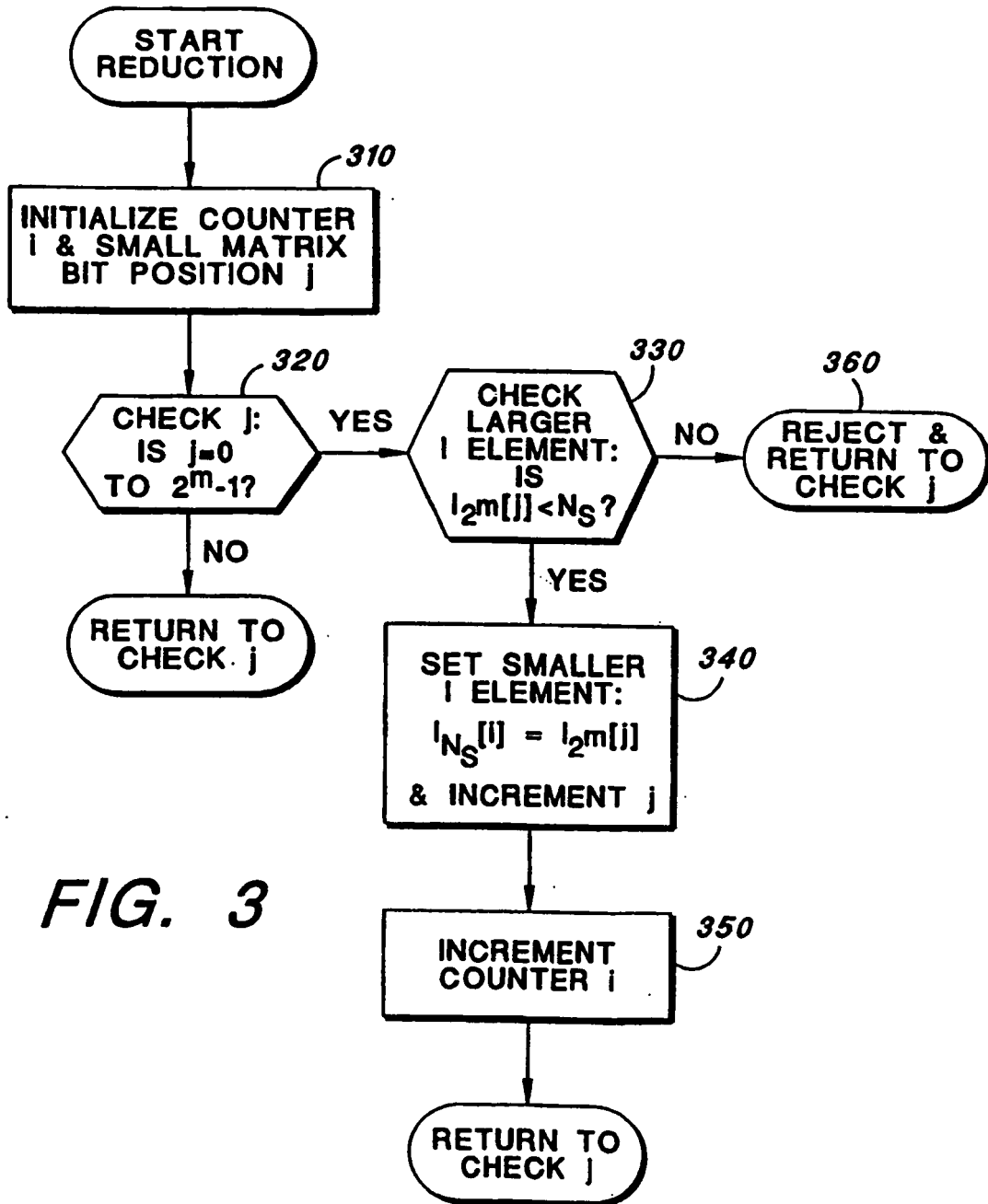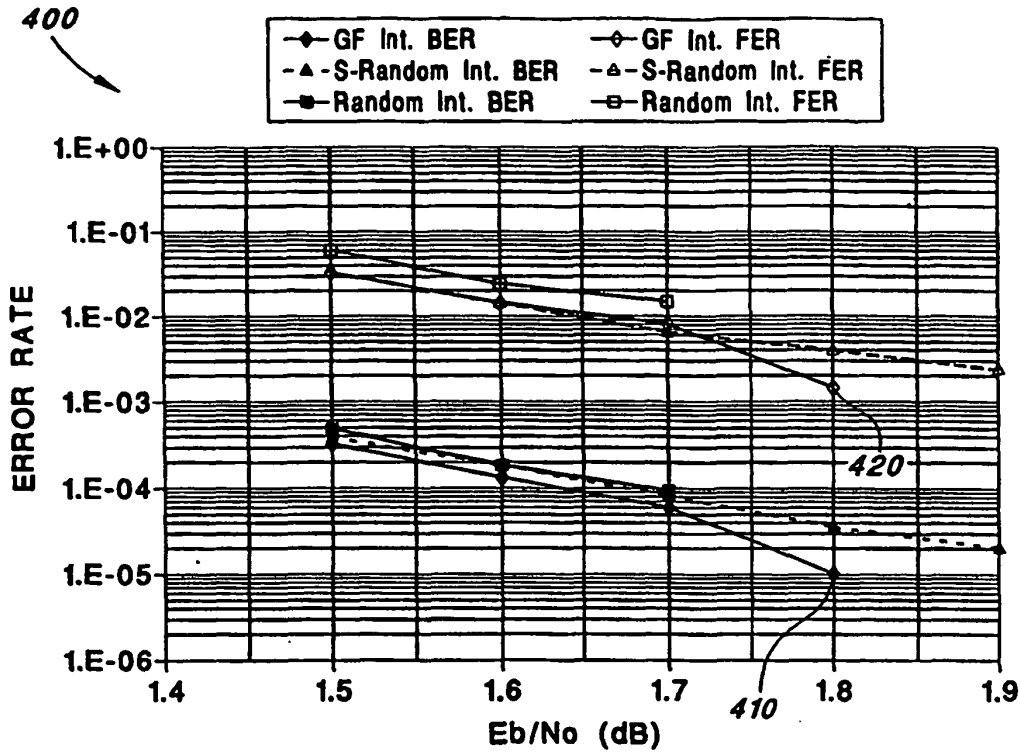FIG. 1

FIG. 2

START
REDUCTION

310

INITIALIZE COUNTER
i & SMALL MATRIX
BIT POSITION j

320

CHECK j:
IS j=0
TO 2^m-1?

YES

330

CHECK
LARGER
i ELEMENT:
IS
$I_2m[j] < N_S$?

NO

360

REJECT &
RETURN TO
CHECK j

NO

RETURN TO
CHECK j

YES

SET SMALLER
i ELEMENT:

$I_{N_S}[i] = I_2m[j]$

& INCREMENT j

340

FIG. 3

INCREMENT
COUNTER i

350

RETURN TO
CHECK j

400

→ GF Int. BER	→ GF Int. FER
-▲-S-Random Int. BER	-▲-S-Random Int. FER
-■-Random Int. BER	-θ-Random Int. FER

420

410

FIG. 4   Performance of Turbo Interleavers of Size: 1024, Code Rate: 1/2, Four Decoder Itereations

500

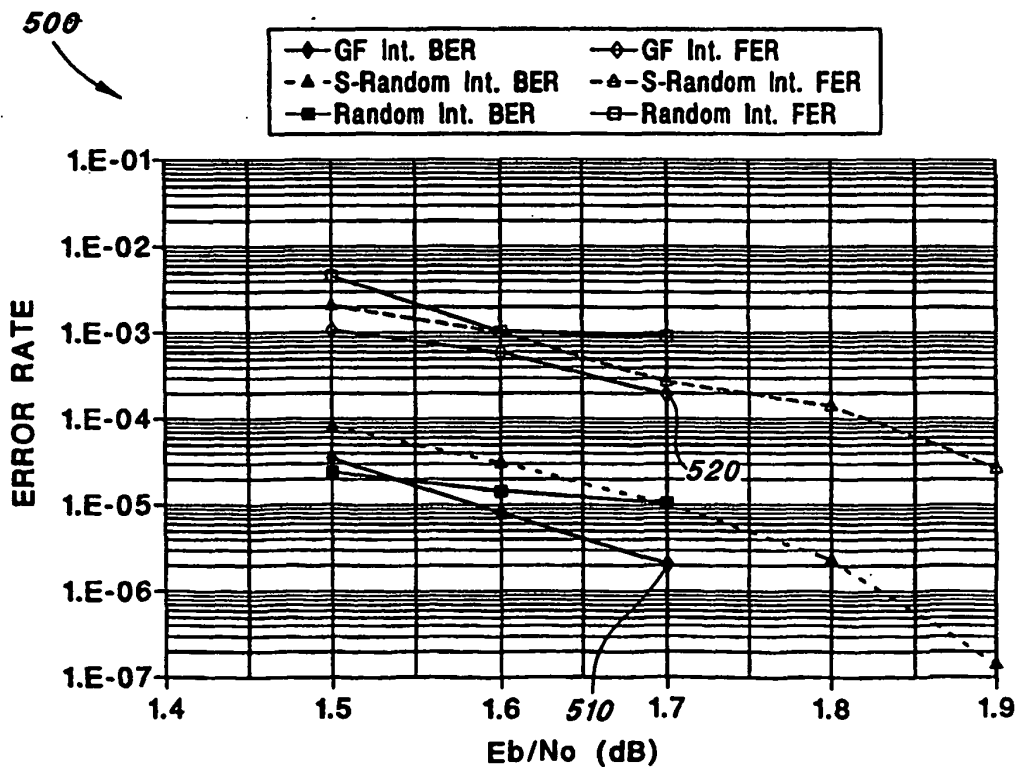|	
→ GF Int. BER	→ GF Int. FER
-▲-S-Random Int. BER	-▲-S-Random Int. FER
-■-Random Int. BER	-θ-Random Int. FER

520

510

FIG. 5   Performance of Turbo Interleavers of Size: 1024, Code Rate: 1/2, Eight Decoder Itereations
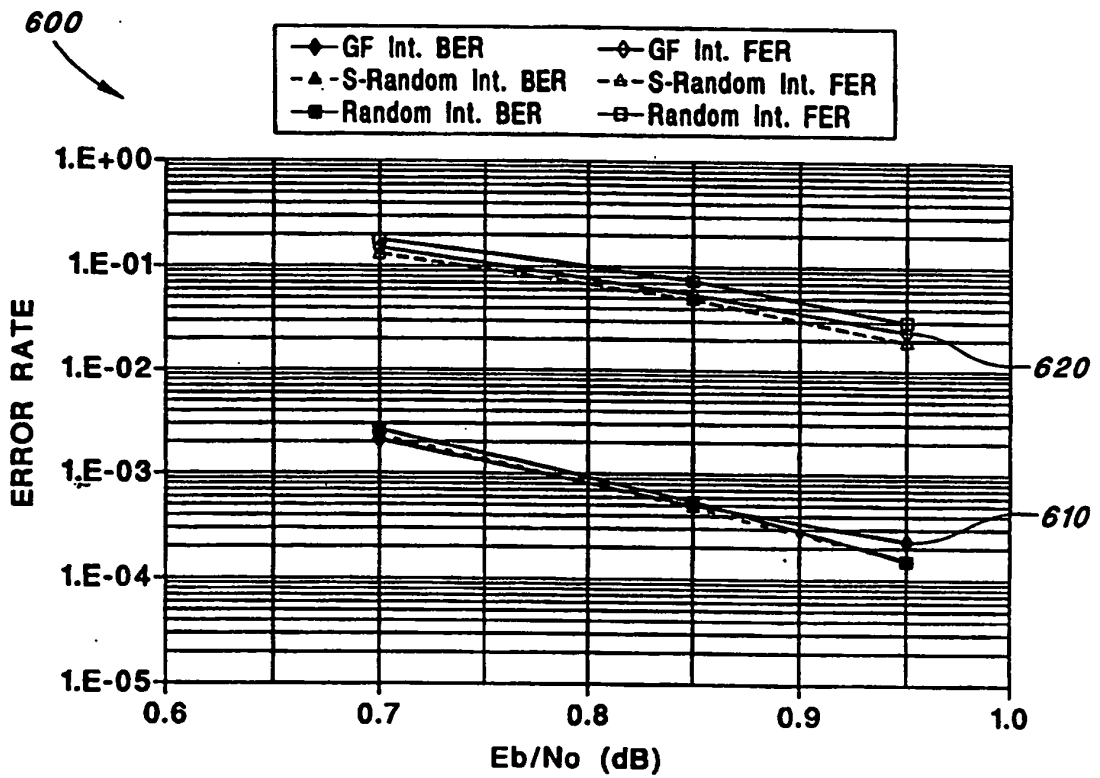
600

FIG. 6 Performance of Turbo Interleavers of Size: 1152, Code Rate: 1/3, Four Decoder Itereations

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Non-patent literature cited in the description

- **D.DIVSALAR ; F. POLLARA.** Turbo Codes for PCS Applications. *proceedings of the international conference on communications, IEEE, Seattle,* June 1995, 54-59 **[0008]**

- **S MARIĆ.** Class of algebraically constructed permutations for use in pseudorandom interleavers. *Electronics Letters,* 18 August 1994, vol. 30 (17 **[0009]**